IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Application of:
Jeffrey M. Ayars
Application No.: 10/075,471
Filed: February 13, 2002

Group Art Unit: 2435
Confirmation No. 7533
Examiner: Paliwal, Yogesh

For: SCALABLE AND EXTENSIBLE SECURE RENDERING OF DIGITAL CONTENT

PRE-APPEAL REQUEST FOR REVIEW

TO THE COMMISSIONER FOR PATENTS:

In response to the Office Action dated January 6, 2009 (hereinafter "Office Action") and the Advisory Action dated March 24, 2009 (hereinafter "Advisory Action), in the above-identified application, Applicant requests review of the final rejection in the above-identified application.

REMARKS/ARGUMENTS

Claims 25-26 were rejected under 35 U.S.C. § 102(e) as being anticipated by *England*. For at least the reasons discussed below, Applicants respectfully submit that *England* fails to disclose every element of Claims 25-26. In particular, Applicants respectfully submit that *England* fails to disclose a root one of a plurality of hierarchically organized digital content rendering modules that performs any of the following elements, as claimed in Claim 25:

- "exclusively receiving with the root one of the… hierarchically organized digital content rendering modules a first digital content of a first type";
- "verifying with a root one of… hierarchically organized digital content rendering modules… [and] re-verifying with said root one of said modules….";
- "rendering in part… re-verifying… and transferring with said root one [module] the first digital content to the re-verified… downstream module";
- verifying with a root content rendering module, that each immediate downstream module has not been compromised, during an initialization period.

***England* does not disclose a root one of a plurality of hierarchically organized digital content rendering modules that performs the operations claimed in Claim 25.**

In the Office Action at 6-7, 10, et al., *England's* Numeral 441 in Fig. 4 is interpreted to be the root one of the… digital content rendering modules. Numeral 441 in Fig. 4 refers to a secure DLL used by a content provider application to store a list of other modules. *E.g.* Col. 8 ln. 16-17; Col. 11 ln. 10-13.

However, Numeral 441 in Fig. 4 is not a "root one of a plurality of hierarchically

organized digital content rendering modules," as claimed. On the contrary, *England* specifically states, "the root of trust of secure code modules is a secure loader 410.... Trust descends hierarchically to one or more security managers 420...." Col. 7 ln. 66-Col. 8 ln. 3. Security manager 420 further passes trust to Numeral 441 in Fig. 4. Col. 8 ln. 25-26. Thus, Numeral 441 in Fig. 4 is at most a middle node in a hierarchy of trust having secure loader 410 as its root. However, as discussed below, *England* never discloses that secure loader 410 is capable of performing any of the other operations (i.e., exclusively receiving, rendering in part, re-verifying, transferring) that are performed by Claim 25's "root one of... digital content rendering modules."

***England* does not disclose "exclusively receiving with the root one of the... hierarchically organized digital content rendering modules a first digital content of a first type."**

England does not disclose a root one of a plurality of hierarchically organized digital content rendering modules to "exclusively receiv[e]... a first digital content of a first type," as claimed. The Office Action asserts that Numeral 441 in Fig. 4 discloses this element, citing to Col. 11 ln. 6-31. However, that portion of *England* discloses merely that "[a]nother function of **security manager 420** is to offer several types of cryptographic services to the [content provider] modules. .... Upon initial load--or later-- the **[security manager 420]** identifies the trusted modules..." (emphasis added). Thus, *England* does not support the Office Action's assertion that Numeral 441 in Fig. 4 discloses a root one of a plurality of hierarchically organized digital content rendering modules to "exclusively receiv[e]... a first digital content of a first type," as claimed.

In addition, neither the cited section, nor any other section of *England* discloses a root one digital content rendering module exclusively receiving digital content of a first type, as claimed. On the contrary, the cited portion of *England* discloses at most that a content provider application ("CPA") may exclusively decrypt a secret key or other data encrypted by a content provider. Col. 11 ln. 19-22. Alternatively, a CPA may exclusively receive a key or other secret data by decrypting a digest with a key provided by security manager 420. Col. 11 ln. 25-28.

Indeed, far from teaching that a root module exclusively receives a first digital content of a first type, *England* actually teaches almost the exact opposite. At Col. 8 ln. 54-58, *England* discloses that "[a]nyone can write an application that chooses not to use the secure path to device driver 460." Thus, *England* fails to teach that a root module **exclusively** receives first content, as claimed. Rather, according to *England*, any module can receive a digital content of a first type, but only certain modules can obtain secret keys to decrypt encrypted content. *See generally* Cols. 9-10.

Thus, Applicants respectfully submit that *England* does not disclose "exclusively receiving with the root one of the plurality of hierarchically organized digital content rendering modules a first digital content of a first type," as claimed in Claim 25.

***England* does not disclose "verifying with a root one of… hierarchically organized digital content rendering modules… [and] re-verifying with said root one of said modules…."**

In addition, Applicants respectfully submit that the Office Action errantly relies on *England's* teachings related to "identification" when Claim 25 is directed to "verification" and then "re-verification." Indeed, *England* teaches a substantive difference between "identifying" a trusted module and "verifying" a trusted module, and *England* fails to teach the claimed "verification" and then "re-verification."

As disclosed at Col. 11 ln. 9-14, trusted modules may be "identified" by reading a list of public keys from a content provider secure DLL either upon initial load or later. However, *England* teaches that actual "verification" of the trusted modules occurs only once. Applicants respectfully point out *England's* discussion of blocks 540 in Figure 5 at Col. 14 ln. 35-45.

> Blocks 540 show the operation of hierarchical trust…. If [an untrusted] module calls a trusted module, block 542 starts trusted interrupt handler 422…. If the security manager names the called module as trusted, **block 543 causes block 544 to verify that its signature is correct** i.e., that it is the module it claims to be.

Thus, *England* discloses that a module may be identified as trusted at initialization or later, but that the module's identity must be verified one time when the module is called (i.e., only when the module is called must it be "verified" to ensure that the module actually is the module it has been identified as). *England* never discloses that a module is "re-verified," let alone "re-verifying with said root one of said modules that one of the [downstream modules] is uncompromised; and transferring with said root one of said modules the first digital content to the re-verified immediate downstream module to further the rendering of the first digital content," as claimed.

This conclusion is further supported by *England's* published claims 38-40 and 43-44, which disclose a module identification step ("determining that the second module is a trusted module") and a single module verification step ("verifying a signature of the second module") with no additional re-verification step.

***England* does not disclose "rendering in part… re-verifying… and transferring with said root one [module] the first digital content to the re-verified… downstream module."**

The Office Action, asserts that *England* discloses rendering in part a first digital content with a root one of the plurality of hierarchically organized digital content rendering modules, re-

verifying a downstream module with the root module, and transferring the first digital content to the re-verified downstream module, as claimed. The Office Action, at 3 and 10, relies on Col. 11 ln. 9-14 to support this assertion. In particular, the Office Action's support for its assertion comes from *England's* teaching that "Upon initial load--or later--the [security manager 420] identifies the trusted modules, so that a call to them maps their pages." According to the Office Action's interpretation, set out on page 3, this statement means that two separate verification steps are performed by the security manager: once during initial load and "once the audio content is downloaded SM once again verify a downstream module[]."

However, Applicants respectfully submit that neither the cited portions of *England*, nor any other portion of *England*, support the proposition for which they are cited. The first paragraph of Col. 11 discloses that a content provider application may name other modules that it trusts by naming a public key of the trusted modules in secure DLL (Numeral 441 in Fig. 4). *England* teaches that the security manager 420 identifies the named modules either upon initial load or at a later time. *England* simply does not teach that security manager 420 identifies the named modules on two separate occasions, as asserted in the Office Action. Thus, the Office Action relies on an unsupported assertion that security manager 420 identifies trusted modules at initial load and once again verifies once audio content is loaded.

Moreover, even assuming for the sake of argument that the Office Action's assertion was supported, *England* nonetheless would not disclose rendering in part a first digital content with a root one of the plurality of hierarchically organized digital content rendering modules, re-verifying a downstream module with the root module, and transferring the first digital content to the re-verified downstream module, as claimed. As admitted in the Office Action, Col. 11 ln. 9-14 describes actions performed by security manager 420. However, according to Claim 25, the claimed steps are performed by a root content rendering module, and the Office Action, at pages 6-7 and 10, interprets Numeral 441 in Fig. 4, not security manager 420, to be a root digital content rendering module. Thus, even according to the Office Action's own interpretation, *England* fails to disclose rendering in part and re-verifying by the root one of the plurality of hierarchically organized digital content rendering modules, as claimed.

***England* does not disclose verifying with a root content rendering module, that each immediate downstream module has not been compromised, during an initialization period.**

Moreover, the Office Action at 3 and 10 states that the claimed "verifying" step is disclosed by *England* at Col. 11 ln. 9-14. However, as discussed above, that portion of *England* discloses merely that "Upon initial load--or later-- **the [security manager 420]** identifies the

trusted modules..." Thus, *England* does not support the Office Action's assertion that Numeral 441 in Fig. 4 "verifies... that each [immediate downstream module] has not been compromised, during an initialization period," as claimed.

For at least the reasons discussed above, Applicants respectfully submit that the Office Action erred in rejecting Claim 25. The Office Action's rejection of Claim 26 is similarly deficient according to similar reasoning.

Claims 27-28; 1-24 and 29-33

Claims 1-24 and 27-33 were rejected under 35 U.S.C. § 103(a) as being unpatentable over either *England* alone or *England* in view of *Graunke*. Applicants respectfully submit that *Graunke* fails to remedy any of the defects in *England* as discussed above. On the contrary, *Graunke* discloses merely a method of securely distributing data to a remote system by means of a tamper resistant key module including a private key of an asymmetric key pair and encrypted data. Accordingly, Applicants respectfully submit that the Office Action erred in rejecting Claims 1-24 and 27-33.

CONCLUSION

For at least the reasons above, Applicants respectfully submit that all pending claims are allowable and request that these claims be allowed to proceed to issuance. We believe the appropriate fees accompany this transmission. If, however, insufficient fee payment or fee overpayment occurs, the amount may be withdrawn or deposited from/to AXIOS Law Group's deposit account. The deposit account number is 50-4051.

Respectfully submitted,
AXIOS LAW GROUP

Date: April 6, 2009

by: /Adam L.K. Philipp/

Adam L.K. Philipp - Reg. No.: 42,071
Direct: 206.217.2226
E-mail: adam@axioslaw.com

AXIOS Law Group
1525 4th Avenue, Suite 800
Seattle, WA 98101
Telephone: 206-217-2200
Customer No.: 61,857